

REMARKS/ARGUMENTS

Claims 1-29 are pending in the present application. Claims 1, 3, 4, 6, 7, 13, 14, 16, 17, 19, 20, 26, and 27 are amended. The claim amendments do not affect the scope or patentability of the claims. Support for the amendments can be found in the originally filed claims and in the specification on page 42, lines 16-18. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 103, Obviousness: Claims 1, 2, 4, 5, 8-15, and 17-28

The examiner rejected claims 1, 2, 4, 5, 8-15, and 17-28 under 35 U.S.C. § 103 as obvious over *Shupak*, Debug annotations, U.S. Patent No. 6,874,140 (March 29, 2005) (hereinafter “*Shupak*”) and *Charisius et al.*, Methods and Systems for Finding Specific Line of Source Code, U.S. Patent No. 6,983,446 (January 3, 2006) (hereinafter “*Charisius*”). This rejection is respectfully traversed. With regard to claim 1, the Examiner states:

Regarding claim 1, Shupak discloses a method comprising:
receiving a designation of a particular portion of a body of source code (Fig. 2, 250, see annotation information and all associated text);
receiving a designation of an association of the particular portion of the body of source code with commentary related to the particular portion of the body of source code (Fig. 3, 310 and all associated text);
receiving a designation of a set of intended readers of the commentary (Fig. 8, 860 also see Fig. 9, 920 and all associated text). Shupak doesn't expressly disclose making the commentary and the associated particular portion of the body of source code accessible to only the set of intended readers.

However, Charisius in an analogous art and similar configuration discloses that private and public attributes, which can read by a reader of the source code regarding how the attributes should be used (32:5-10) also see 17:15-25, which shows protected and public classes).

Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine Shupak and Charisius because, it would enable controlling access by the reader to certain classes or source instruction portions.

Office Action dated January 4, 2007, pp. 2-3.

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). A *prima facie* case of obviousness is established when the teachings of the prior art itself suggest the claimed subject matter to a person of ordinary skill in the art. *In re Bell*, 991 F.2d 781, 783, 26 U.S.P.Q.2d 1529, 1531 (Fed. Cir. 1993). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). For an invention to be *prima facie* obvious, the prior art must teach or suggest all claim limitations. *In re Royka*,

490 F.2d 981, 180 USPQ 580 (CCPA 1974). In the case at hand, the cited references when considered as a whole do not teach or suggest all of the limitations of the claims, arranged as they are in the claims.

I.A. The Proposed Combination Does Not Teach all of the Features of Claim 1

Claim 1 as amended recites:

1. A method comprising:
receiving a first designation of a particular portion of a body of source code;
receiving a second designation of an association of the particular portion of the body of source code with commentary related to the particular portion of the body of source code;
receiving a third designation of a set of intended readers of the commentary; and
making the commentary and the particular portion of the body of source code accessible to only the set of intended readers.

Despite the Examiner's assertion otherwise, *Shupak* in view of *Charisius* does not obviate claim

1. *Shupak* in view of *Charisius* does not teach or suggest either of the claimed features of, "receiving a designation of a set of intended readers of the commentary", or "making the commentary and the particular portion of the body of source code accessible to only the set of intended readers," as required in claim 1.

Shupak discloses an object code compiler provided with an intrinsic annotation function that provides quasi-machine language comments to the machine code during a compiling process. *Shupak's* analysis tools, such as debuggers and the like, are adapted to read the quasi-machine language comments. The comments are then used to override or change the normal behavior of the analysis tools. Specifically, *Shupak* teaches:

An intrinsic function is supported by a compiler to annotate the program code, so that analysis tools, such as debuggers and profilers, have more information with which to analyze performance or whatever is of interest. The annotation information in the executable program remains associated with the code that the annotation function is associated with. More specifically, an annotation function in the source code is parsed and the parsed function is transformed into an intermediate form, and annotation information is generated from the intermediate language code. Therefore, the annotation information corresponds to the annotation function. In another embodiment, a new type of syntax specific to annotations is created and a compiler is updated or modified to support the syntax.

Analysis tools are adapted to respond to the annotation information by changing the execution of the computer program analysis tool using the annotation, such as using the annotation to override the default behavior of the

tool. This eliminates the need to create input command files for the analysis tools, and the consequent need to update the input command files in parallel to the source code.

Shupak, col.3, ll. 47-67.

With regard to the claim feature of “receiving a set of intended readers of the commentary,” the Examiner cites “Fig. 8, 860 also see Fig. 9, 920 and all associated text.” Both elements 860 and 920 of *Shupak* disclose a reader of this “intermediate form” quasi-machine language. *Shupak*’s Element 860 is described by *Shupak* as follows:

a receiver 860 operably coupled to the front-end component 820, that receives the parsed annotation function 840. The back-end component 850 also has a transformer 870 that is operably coupled to the receiver 860, that transforms the parsed annotation function 840 into intermediate language code 880.

Shupak, col.11, ll. 62-67.

Likewise, *Shupak*’s Element 920 is described by *Shupak* as follows:

a receiver or reader 920 of annotation information in an executable computer program 910 during execution of the executable computer program. The annotation information having been generated from an intrinsic annotation function that the executable computer program was compiled from, as in methods 300, 400, 500, 600, and 700. The annotation information includes information, such as a symbolic address and one or more operands (not shown).

The analysis tool apparatus 900 includes an execution controller 930 that controls the execution of the computer program analysis tool using the annotation information. In one embodiment, the annotation information overrides the default behavior of the tool. The execution controller 930 is operably coupled to the receiver 920.

Shupak, col.12, ll. 14-27.

As indicated by these portions of *Shupak*, *Shupak*’s reader does not disclose the claimed method’s “receiving a set of intended readers of the commentary”. *Shupak*’s readers 860 and 920 are fixed parts of the compiler/analysis tool capable of converting the quasi-machine code into executable instructions. While *Shupak*’s reader may be responsible for reading annotations, nothing is input into *Shupak* that would cause any feature of *Shupak* to receive *Shupak*’s reader, let alone receive a set of *Shupak*’s readers. In contrast, readers of the presently claimed invention are a listing of potential persons who may view changes to a source code file made by a developer. The reader of *Shupak* is therefore not analogous to the set of intended readers in claim 1. Because *Shupak* does not teach this claimed feature, and the Examiner does not cite any equivalent feature in *Charisius*, *Shupak* in view of *Charisius* does not obviate claim 1 under the standards of *In re Lowry*.

Additionally, the Examiner admits that *Shupak* does not disclose the feature of, “making the commentary and the particular portion of the body of source code accessible to only the set of intended readers”, and instead relies on *Charisius* to disclose this deficiency. *Charisius* discloses:

[an] improved software development tool of the present invention [that] allows a developer to simultaneously view a graphical and a textual display of source code. The graphical and textual views are synchronized so that a modification in one view is automatically reflected in the other view. In addition, the software development tool is designed for use with more than one programming language.

Charisius, col. 3, ll. 14-20.

For support of the current rejection, the Examiner cites the following sections of *Charisius*:

Private attributes that never get their values changed must be declared final. By explicitly declaring them in such a way, a reader of the source code gets some information regarding how the attribute should be used. Thus, in the following source code:

```
class CPAMBF {  
    int attr1 = 10;  
    int attr2 = 20;  
    void func () {  
        attr1 = attr2;  
        System.out.println(attr1);  
    }  
}
```

all private attributes that never change should be made final, as follows:

```
class CPAMBF {  
    int attr1 = 10;  
    final int attr2 = 20;  
    void func () {  
        attr1 = attr2;  
        System.out.println(attr1);  
    }  
}
```

Charisius, col. 32, ll. 7-35.

List All Public And Package Members
First

Order of Class Members Declaration

Enforces a standard to improve readability. Methods/data in your class should be ordered properly. According to Sun Code Conventions for Java, the parts of a class or interface

declaration should appear in the following order

1. Class (static) variables. First the public class variables, then the protected, then package level (no access modifier), and then the private.

2. Instance variables. First the public class variables, then the protected, then package level (no access modifier), and then the private.

3. Constructors

4. Methods Order Of Checks for correct ordering of modifiers.

Charistus, col. 17, ll. 13-25.

These cited statements disclose a preferred order for introducing public and private variables into source code, and the code for introducing those public and private variables. The cited statements do not teach anything regarding the inclusion of commentary, let alone commentary wherein “the commentary and the particular portion of the body of source code [are] accessible to only the set of intended readers.”

Because *Shupak* in view of *Charistus* does not teach or suggest either “receiving a designation of a set of intended readers of the commentary” or “making the commentary and the particular portion of the body of source code accessible to only the set of intended readers” as required in claim 1, *Shupak* in view of *Charistus* does not obviate claim 1. Therefore, the rejection of claim 1 under 35 U.S.C. § 103 has been overcome.

Claims 14 and 27 also contain features similar to those presented in claim 1. Therefore, *Shupak* in view of *Charistus* does not obviate these claims at least for the reasons presented above. Therefore, the rejection of claim 14 and 27 under 35 U.S.C. § 103 has been overcome.

Claims 2, 4, 5, and 8-13 depend from claim 1. Claims 15 and 17-26 depend from claim 14. Claim 28 depends from claim 27. Therefore, the same distinctions between *Shupak* in view of *Charistus* and claims 1, 14, and 27 apply to these dependent claims. Consequently, the rejection of claims 2, 4, 5, 8-13, 15, 17-26 and 28 under 35 U.S.C. § 103 has been overcome.

I.B. No Motivation Exists to Combine *Shupak* and *Charistus* Because They Address Different Problems

Additionally, the examiner failed to state a *prima facie* obviousness rejection because no teaching, suggestion, or motivation exists to combine the references in the manner proposed by the examiner. No teaching, suggestion, or motivation exists because the references are directed towards solving different problems.

It is necessary to consider the reality of the circumstances--in other words, common sense--in deciding in which fields a person of ordinary skill would reasonably be expected to look for a solution to the problem facing the inventor. *In re Oetiker*, 977 F.2d 1443 (Fed. Cir. 1992); *In re Wood*, 599 F.2d 1032, 1036, 202 U.S.P.Q. 171, 174 (CCPA 1979). In the case at hand, the cited references address distinct problems. Thus, no common sense reason exists to establish that one of ordinary skill would reasonably be

expected to look for a solution to the problem facing the inventor. Accordingly, no teaching, suggestion, or motivation exists to combine the references and the examiner has failed to state a *prima facie* obviousness rejection of claim 1.

For example, *Shupak* is directed to the problem using a code compiler provided with an intrinsic annotation function to provide quasi-machine language comments to the machine code during a compiling process. *Shupak*'s analysis tools, such as debuggers and the like, are adapted to read the quasi-machine language comments. The comments are then used to override or change the normal behavior of the analysis tools. For example, *Shupak* provides that:

Previous systems which provided spoken messages to be transmitted between two computer across a local area network (LAN) were not able to deliver the spoken messages in real-time. Rather, the previous systems operated in a batch mode capacity. These type of voice communication systems recorded an entire spoken message and then play that entire spoken message back at the receiver, although with a substantial time delay between the recording of the message and the playing back.

Baugh, col. 1, ll. 18-26.

On the other hand, *Charisius* is directed to the problem of simultaneously viewing a graphical and a textual display of source code. A graphical representation or model of the source code is helpful to organize and visualize the structure and components of the system. The graphical and textual views are synchronized so that a modification in one view is automatically reflected in the other view. For example, *Charisius* provides as follows:

[an] improved software development tool of the present invention [that] allows a developer to simultaneously view a graphical and a textual display of source code. The graphical and textual views are synchronized so that a modification in one view is automatically reflected in the other view. In addition, the software development tool is designed for use with more than one programming language.

Charisius, col. 3, ll. 14-20.

Based on the plain disclosures of the references themselves, the references address completely distinct problems that are unrelated to each other. The problem of using a code compiler to provide quasi-machine language comments to the machine code intrinsic during a compiling process is completely distinct from simultaneously viewing a graphical and textual display of source code to organize and visualize the structure and components of the system.

Because the references address completely distinct problems, one of ordinary skill would have no reason to combine or otherwise modify the references to achieve the invention of claim 1. Thus, no

proper teaching, suggestion, or motivation exists to combine the references in the manner suggested by the examiner. Accordingly, the examiner has failed to state a *prima facie* obviousness rejection against claim 1 or any other claim in this grouping of claims.

II. 35 U.S.C. § 103, Obviousness: Claims 3, 16 and 29

The examiner rejected claims 3, 16 and 29 under 35 U.S.C. § 103 as obvious over *Shupak* and *Charisus* as applied to Claim 1, 14 and 27 in view of *Kay*, Method of Designing an Integrated Circuit Using Scheduling and Allocation with Parallelism and Handshaking Communication, and an Integrated Circuit Designed by Such Method, U.S. Patent No. 6,021,266 (February 1, 2000) (hereinafter “*Kay*”). This rejection is respectfully traversed.

In reference to Claims 3 and 16, the Examiner states:

Regarding claims 3 and 16, *Shupak* as modified with *Charisus* discloses all the claimed limitations as applied in claims 1, 14 and 27. The combination of *Shupak* and *Charisus* doesn't expressly disclose receiving a designation of a particular time period during which the commentary will be available, wherein the commentary and associated particular portion of the body of source code are accessible to the set of intended readers during the particular time period. However, *Kay* in an analogous art and similar configuration discloses that a user may provide timing information by annotating clock edges and also discloses scheduling (2:40 - 45). Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine *Shupak*, *Charisus* and *Kay* because, it would enable providing timed information.

Office Action dated January 4, 2007, p. 7.

Claims 3, 16, and 29 are dependent from claims 1, 14, and 27 respectively. Furthermore, *Kay* does not overcome the stated deficiencies of *Shupak* in view of *Charisus*. Therefore, the same distinctions between *Shupak* in view of *Charisus* and claims 1, 14, and 27 as stated above apply to these dependent claims. Additionally, and contrary to the Examiner's assertion, *Kay* does not teach the feature of “receiving a designation of a particular time period during which the commentary will be available, wherein the commentary and associated particular portion of the body of source code are accessible to the set of intended readers during the particular time period.”

Specifically, *Kay* teaches:

a method of designing an integrated circuit comprising defining the functions of the integrated circuit in a programming language supporting parallelism and synchronized communication, and applying a compiler which is arranged to retime synchronized communications without changing the order of external communications of the integrated circuit so as to produce output code representing circuitry of the integrated circuit.

Kay, col. 3, ll. 46-54.

With regard to the claim feature of “receiving a designation of a particular time period during which the commentary will be available, wherein the commentary and particular portion of the body of source code are accessible to the set of intended readers during the particular time period”, the Examiner cites *Kay*, column 2, lines 40-45. This section states:

The compiler may supply optimizations to trade off speed and area by means of scheduling and allocation style algorithms as disclosed in Reference 2.

The user must still provide timing information by annotating where clock edges are to occur and must know on which clock cycles input and output data must be available.

Kay, col. 2, ll. 40-45.

This passage is a generic statement about clock cycles in microprocessors that, in the specific case of *Kay*, are used to determine speed parameters of designed circuits. Nothing in the cited passage teaches or suggests the feature of “receiving a designation of a particular time period during which the commentary will be available, wherein the commentary and particular portion of the body of source code are accessible to the set of intended readers during the particular time period.”

Because *Shupak* in view of *Charisius* and *Kay* does not teach or suggest “receiving a designation of a particular time period during which the commentary will be available, wherein the commentary and particular portion of the body of source code are accessible to the set of intended readers during the particular time period” as required in claim 3, *Shupak* in view of *Charisius* and *Kay* does not obviate claim 3. Therefore, the rejection of claim 3 under 35 U.S.C. § 103 has been overcome.

Claims 16 and 29 also contain features similar to those presented in claim 3. Therefore, *Shupak* in view of *Charisius* and *Kay* does not obviate these claims at least for the reasons presented above. Therefore, the rejection of claims 16 and 29 under 35 U.S.C. § 103 has been overcome.

III. **35 U.S.C. § 103, Obviousness: Claims 6 and 19**

The examiner has rejected claims 6 and 19 under 35 U.S.C. § 103 as being unpatentable over *Shupak* and *Charisius* as applied in claims 4 and 17 and further in view of *Ben-Romdhane et al.*, System and Method for Manipulation of Software, U.S. Patent Publication No. 2004/0031015 A1 (February 12, 2004) (hereinafter “*Ben-Romdhane*”). This rejection is respectfully traversed.

In reference to Claims 6 and 19, the Examiner states:

Regarding claims 6 and 19, *Shupak* as modified with *Charisius* discloses all the claimed limitations as applied in claims 4 and 17. The combination of *Shupak* and *Charisius* doesn't expressly disclose wherein the notification is sent to the at least a subset of the set of intended readers via electronic mail. However *Ben-Romdhane* discloses a source editor which includes source annotation and also a notification trigger (e.g. email)

which indicates when one or more files have been modified (0193). Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine Shupak, Charisus and Romdhane because it would enable notifying users when source code has been modified.

Office Action dated January 4, 2007, p. 8.

Claim 6 is dependent from claim 1. *Ben-Romdhane* does not overcome the stated deficiencies of *Shupak* in view of *Charisius*. *Ben-Romdhane* discloses a method and system for modeling the constituent components of a software program into a graphical presentation showing data dependencies, functional dependencies, and control flow indicators. While *Ben-Romdhane* does disclose update notifications, *Ben-Romdhane* does not disclose either “receiving a designation of a set of intended readers of the commentary” or “making the commentary and the particular portion of the body of source code accessible to only the set of intended readers” as required by virtue of the dependency of claim 6 from claim 1.

Therefore, the same distinctions between *Shupak* in view of *Charisius* and claim 1 as stated above apply to claim 6. Because *Shupak* in view of *Charisius* and *Ben-Romdhane* does not teach or suggest either of “receiving a designation of a set of intended readers of the commentary” or “making the commentary and the particular portion of the body of source code accessible to only the set of intended readers” as required by virtue of the dependency of claim 6 from claim 1, *Shupak* in view of *Charisius* and *Ben-Romdhane* does not obviate claim 6. Therefore, the rejection of claim 7 under 35 U.S.C. § 103 has been overcome.

Claim 20 also contain features similar to those presented in claim 6. Therefore, *Shupak* in view of *Charisius* and *Ben-Romdhane* does not obviate claim 19 at least for the reasons presented above. Therefore, the rejection of claim 19 under 35 U.S.C. § 103 has been overcome.

Therefore, the rejection of claims 6 and 19 under 35 U.S.C. § 103 has been overcome.

IV. 35 U.S.C. § 103, Obviousness: Claims 7 and 20

The examiner has rejected claims 7 and 20 under 35 U.S.C. § 103 as being unpatentable over *Shupak* and *Charisius* as applied to claims 1 and 14 and further in view of *Stern, System and Method for Automated Annotation of Files*, U.S. Patent No. 6,572,661 (June 3, 2003) (hereinafter “*Stern*”). This rejection is respectfully traversed.

In reference to Claim 7, the Examiner states:

Regarding claim 7, *Shupak* as modified with *Charisius* discloses all the claimed limitations as applied in claim 1. The combination of *Shupak* and *Charisius* doesn't expressly disclose wherein the designation of an association of the particular portion of the body of source code with commentary related to the particular portion of the body of source code is represented in the form of at least one tag in a markup language, wherein the at least one tag is embedded within the body of source code. However, *Stern* discloses

in an analogous art of source code annotating the inclusions of tags in a markup language wherein the tag is embedded in the source code (FIG. 3,312,314, 322, and all associated text).

Office Action dated January 4, 2007, pp. 8-9.

Claim 7 is dependent from claim 1. *Stern* does not overcome the stated deficiencies of *Shupak* in view of *Charisius*. *Stern* discloses a method and apparatus for the creation of HTML annotated source and comment files, for the purpose of providing a text based code walkthrough. *Stern* does not disclose either “receiving a designation of a set of intended readers of the commentary” or “making the commentary and the particular portion of the body of source code accessible to only the set of intended readers” as required by virtue of the dependency of claim 7 from claim 1.

Therefore, the same distinctions between *Shupak* in view of *Charisius* and claim 1 as stated above apply to claim 7. Because *Shupak* in view of *Charisius* and *Stern* does not teach or suggest either of “receiving a designation of a set of intended readers of the commentary” or “making the commentary and the particular portion of the body of source code accessible to only the set of intended readers” as required by virtue of the dependency of claim 7 from claim 1, *Shupak* in view of *Charisius* and *Stern* does not obviate claim 7. Therefore, the rejection of claim 7 under 35 U.S.C. § 103 has been overcome.

Claim 20 also contain features similar to those presented in claim 7. Therefore, *Shupak* in view of *Charisius* and *Stern* does not obviate claim 20 at least for the reasons presented above. Therefore, the rejection of claim 20 under 35 U.S.C. § 103 has been overcome.

V. **Conclusion**

The subject application is patentable over the cited references and should now be in condition for allowance. The examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: April 9, 2007

Respectfully submitted,

/Theodore D. Fay III/

Theodore D. Fay III
Reg. No. 48,504
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants

TF/BW